EXHIBIT 2

REDACTED

EXHIBIT 3

REDACTED

EXHIBIT 4

REDACTED

EXHIBIT 5

REDACTED

EXHIBIT 6

MONTECH'87 • Conférence sur les Communications
MONTECH'87 • Conference on Communications

DESIGN ARCHITECTURES OF A DTDM PACKET ASSEMBLER AND PACKET MULTIPLEXER

H. J. Chao

Bell Communications Research, Inc.
435 South Street
Morristown, New Jersey, USA

ABSTRACT

Dynamic Time Division Multiplexing (DTDM) combines the concepts of conventional time division multiplexing and packet transmission. It provides the flexibility to satisfy broadband services' variable bit rates, ranging from low bit-rate data to full motion video, and a practical migration strategy from the existing predominately circuit switched network to the future broadband packet switched network. In this paper, we describe a simple modularized Packet Assembler to multiplex different broadband services into a DTDM format serial bit stream. We then present a novel multiple-input-multiple-output statistical Packet Multiplexer concentrating network traffic for more efficient use of transmission facilities. A CMOS VLSI chip designed for used in the Packet Assembler and Packet Multiplexer is discussed at the end.

1. Introduction

Dynamic Time Division Multiplexing (DTDM)[1] is a flexible network transport technique capable of handling both continuous and bursty traffic effectively. By combining the concept of conventional Time Division Multiplexing (TDM) and packet transmission techniques,[2] [3] DTDM provides a practical migration strategy from the existing predominately circuit switched network to the future broadband packet switched network. As illustrated in Fig. 1, the transmission bit stream is divided into frames as is a conventional TDM signal. Each frame consists of three fixed length fields - transmission overhead, packet header and information field. The transmission overhead may contain frame synchronization words and control bytes. The packet header is designed to provide information such as packet occupancy (full or empty), channel number, line number, error detection and so on.

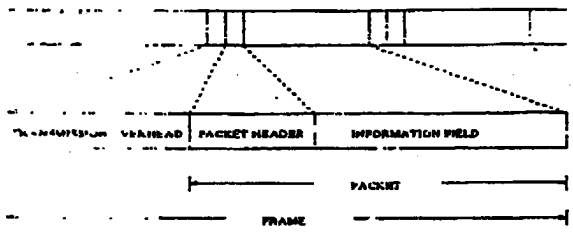

Fig. 1 Dynamic Time Division Multiplex (DTDM) format

A double star architecture has been proposed for a broadband communication system.[4] Single mode optical feeders form a first level star connection from a broadband Central Office (CO) to a Remote Electronics (RE). Distribution and drop fibers form a second level star from the RE to subscribers. Figure 2 shows that different types of services are first multiplexed into a single basic rate DTDM signal using a Packet Assembler in the User-Network Interface (UNI). The DTDM streams are then concentrated into more continuous traffic using a Packet Multiplexer in the RE. The DTDM data stream (slotted packet) is then routed in a packet switch network (e.g. a Batcher Banyan network[5]) in the CO.

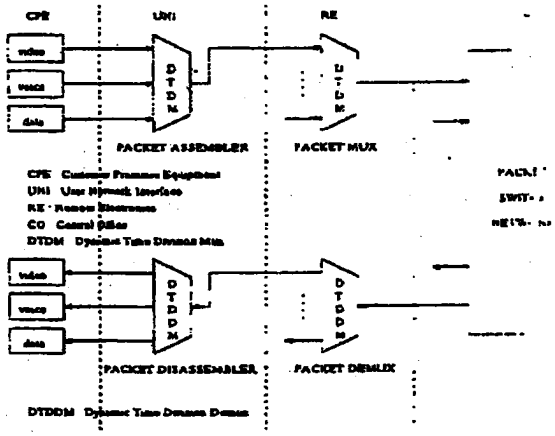

Fig. 2. Transport integration of circuit and packet traffic

In this paper, we describe two multiplexing architectures, which can be implemented using available technology components. The design architectures discussed in this paper are not sensitive to a particular rate and format. In section 2, we describe the design architecture of the Packet Assembler. Section 3 describes a distributed control architecture for a Packet Disassembler. In section 4, by extending earlier work,[6] we present a novel architecture for a multiple-input-multiple-output packet statistical multiplexer. Section 5 describes a Packet Demultiplexer with distributed control. In section 6 an ADROP chip (ADD/DROP a packet to/from a DTDM serial bit stream), a common and crucial component used in two stages of multiplexing, is described in detail. Concluding remarks are given in section 7.

2. Packet Assembler

The function of the Packet Assembler is to packetize the incoming data stream associated with a particular service or transmission channel into the basic DTDM transmission frames. The inputs to the Packet Assembler might have a wide range of bit rates; for example, there could be digital video, voice, and data. Therefore, the Packet Assembler's architecture must be capable of accommodating different bit rates efficiently and be flexible enough to allow future expansion or changing of the input connections to different services. The architecture shown in Fig. 3 provides the capability to add or drop services easily.

Each input service is connected to a Packetizer, which puts the service data into a packet structure by adding a packet header. The packet header carries information such as packet occupancy, channel identification number, line identification number, check sum and so on. The channel identification number is used to identify the packet from one input service versus another. After the data is put into a packet structure, it is stored in a First In-First-Out (FIFO) buffer with byte wide format. The ADROP chip will then conditionally read the data from the FIFO into its parallel data input port (pdi) and generate

CH2517-1/87/0000-0083$01.00©1987 IEEE

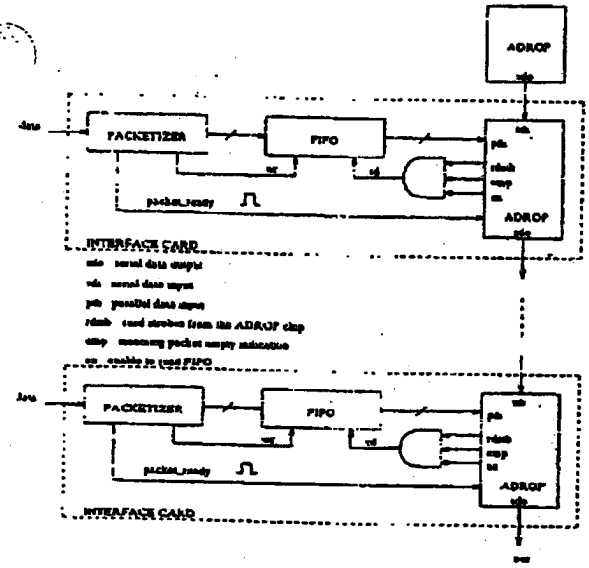

Fig. 3. Packet assembler architecture

a properly framed data stream on its serial data output (sdo) by adding the transmission overhead to the packet.

The ADROP chip will not read the data from the FIFO unless two conditions are met. One is that the "packet_ready" pulse signal from the Packetizer is asserted, indicating one packet is completely stored in the FIFO. The other condition is that the incoming frame on serial data input (sdi) is not already occupied by a valid packet (i.e. an empty frame). The "packet_ready" signal triggers an enable signal, "en", in the ADROP chip to be asserted for the whole frame transmission period. Since the ADROP chips are daisy-chained together, with the higher positions in the figure having higher priority (better service), the contention for empty frame slots is solved automatically by the inherent characteristics of the daisy-chain structure. The ADROP chip examines packet occupancy each time it receives a frame. If the frame is occupied, the "emp" signal will not be asserted and the data won't be read out of the FIFO. On the other hand, if the frame is not occupied, the "emp" signal will be asserted for the entire frame period and the data in the FIFO will be sent out through the serial data output (sdo) if the "en" signal is asserted as well. In the meantime, the packet occupancy indication of the frame is set so that the following ADROPs will not overwrite the frame and it will eventually pass to the network.

In order to simplify the Packet Assembler complexity and hence reduce the building cost, one practical assumption is made : the total traffic of all input services at any given time is less than the information payload of the serial data output. A FIFO with more than one packet capacity guarantees that the packets in the FIFO will never be overwritten because the packet stored in the FIFO will be read out before the next packet arrives. Consequently, there is no need for the ADROP chips to send an acknowledgement signal to the Packetizer. The topmost ADROP chip doesn't have any input service connected to it. It keeps sending empty frames to poll the following ADROP chips. If none of the "packet_ready" signals are asserted, an empty frame is finally sent out through the serial data output (sdo) of the bottom ADROP chip.

3. Packet Disassembler

After the packet has traveled through the entire communication network, which may include packet multiplexers, packet switching networks, and packet demultiplexers, it arrives at a particular service device. The Packet Disassembler, shown in Fig. 4, distributes the incoming frames to the right service devices and removes both transmission overhead and packet header from the frames. The incoming frame is received by each ADROP chip at the input port simultaneously. The packet occupancy and channel identification number are examined by the ADROP chip, which generates proper control signals, "emp" and "match", to determine whether or not the packet will be written into the FIFO.

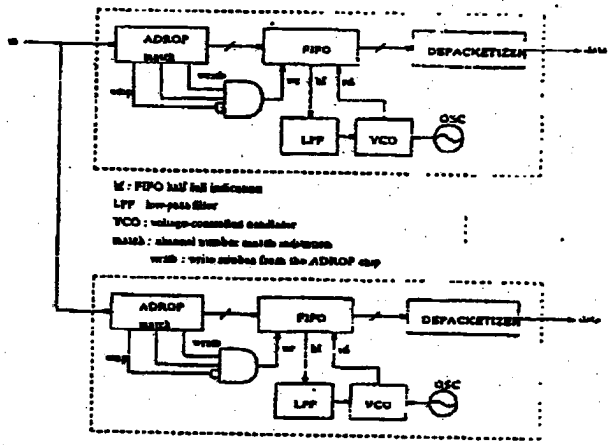

Fig. 4. Packet disassembler with distributed control

Recovering the correct frequency from the incoming data is a very challenging task at the receiving end, because the frequency associated with a particular service may not be carried by the network. Although for each kind of service the typical frequency is known, the difference between the clocks at the transmit and receive ends results in eventual data overwriting or reading false data. There are two ways to control the phase between writing and reading clocks.[7] The first modifies the phase by proper slips during, for example, the silence intervals of sound or the frame intervals of video. The second uses a phase locked loop to modify the reading frequency in order to cancel the frequency difference between the transmitting and receiving clocks. Since the second method is more general and can be used for any kind of service device, it is the one we will describe.

As shown in Fig. 4, a local reading clock (rd) is phase locked with the incoming information rate (wr) so that the data can be read out correctly from the FIFO without overreading or underreading. The packet is written into the FIFO with network clock rate, but read out at a rate dependent on the service. A "hf" signal, indicating the half full of the FIFO, is smoothed out by a low-pass filter (LPF) and is then used to control the output frequency of a voltage-controlled oscillator (VCO). If the read clock (rd) is faster than the input information rate (wr), the "hf" signal will not be asserted. This causes the voltage output from the LPF to decrease, reducing the output frequency from the VCO. If the read clock is slower, the "hf" signal will be asserted and the read clock will be adjusted faster. The same circuit for the Packet Disassembler can be used for different service devices by choosing a proper oscillator frequency.

4. Packet Multiplexer

4.1 N-Input-M-Output Packet Multiplexer

The function of the Packet Multiplexer is to concentrate incoming DTDM streams into more continuous traffic, resulting in more efficient use of the transmission facility. A working prototype, operating at 140 Mb/s with multiple input lines and one output line, is reported in [6]. It was also shown in [6] that both the probability of buffer overflow and the average delay for bursty traffic can be significantly decreased by increasing the number of outputs. Consequently, a novel architecture for a multiple input multiple-output statistical packet multiplexer was designed so that the total traffic load of the multiplexer can be evenly distributed among the output lines and the service priority of each input line can be rearranged dynamically.

The N Input M Output Multiplexer (N:M) Packet multiplexer can be implemented by adding an (N+M) x (N+M) cross point switch network,[8] connecting the serial data output (sdo) of each ADROP to the input of the switch, and connecting the serial data input (sdi) to the output of the switch, as shown in Fig. 5. The connections through the switch network are established by a separate controller.

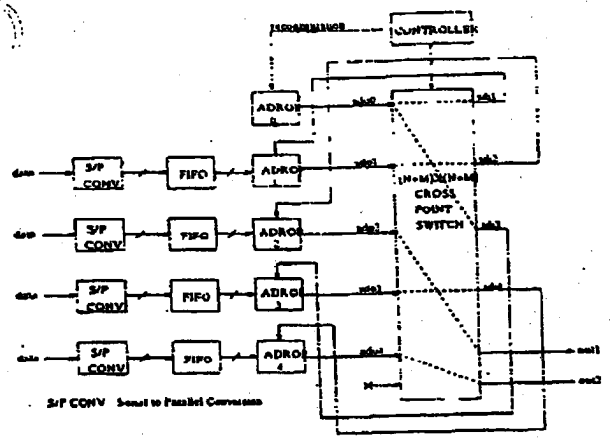

Fig. 5. N:M Packet multiplexer using distributed control (N=4, M=2)

Serial data on each input line is converted into a byte wide stream and stored in the FIFO buffer. The operation of each ADROP chip is similar to that of the Packet Assembler. The ADROP chips won't send out packets stored in the FIFO unless the incoming frame slot is unoccupied. ADROP0 , the head end chip, doesn't have any input lines connected to it. In normal operation, it continuously sends out empty frames to poll input lines. When the system is initialized, the N input lines are divided into M groups, (1, 2, .., J), (J+1, J+2, .., 2J), ..., (MxJ+1, ..., M), where J = N mod M. The J input lines in each group are logically connected as shown in Fig. 6. Their priorities are arranged in a descending order. The line number for each group is denoted as X1, X2, ..., XM. Its sum is equal to N. The line number may change as network reconfiguration occurs for balancing traffic load on the M output lines. The order of input lines in each group may also change according to the priority for input lines. The topmost ADROP chip in each group receives empty frames broadcast from the ADROP0.

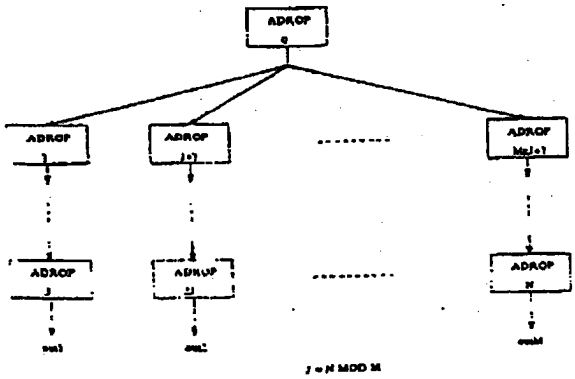

Fig. 6. Logical connection for the M Groups of the N input lines

If one of the ADROPs is not functioning, it can be switched out of the network. Therefore, no packet will be routed through the faulty ADROP and the whole system will continue to work. The cross point switch's broadcast capability allows more than one of the ADROPs to receive empty frames from the ADROP0 at the same time. This achieves the N:M packet multiplexing function automatically and elegantly.

If input lines are not grouped with even traffic, they can be regrouped easily by changing the switch network connections. For example, line J in the first group may be assigned to any other group, say the second group. Then the X1 value decreases by one while X2 increases by one. To spread out traffic evenly among the output lines, the controller must know the traffic statistics of each input and output line and follow some algorithm to rearrange the connection between the inputs and outputs. A conceivable problem arises when an occupied frame is being transmitted between ADROPs at the same time that the switch connections are changed. This results in a valid packet being damaged.

One way to eliminate the problem is that when the controller wants to change the network connection, it informs ADROP0 with a "reconfiguration" signal to force it to send occupied frames instead of empty frames. Therefore, the following ADROPs receive occupied frames and won't send out any valid packet from the associated FIFO. The controller can then change the connections without corrupting any valid packet. During the reconfiguration, all ADROPs except ADROP0 will lose synchronization because of the failure to receive frame bytes at the expected time intervals. When out of frame, it takes several packet times (for example three) to get resynchronized. Therefore, it might be necessary for the ADROP0 to continue sending out occupied frames (for example 10 frames) to cover the time needed for setting up the cross point connections and the time needed for reframing. On other hand, if the cross point switch network is synchronized with a common clock (the data signal is regenerated in each cross point chip),[9] and if the network reconfiguration occurs at the proper time such that no frame synchronization word is destroyed, the ADROP chips won't lose synchronization. Hence, no reframing is necessary and the number of occupied frames the ADROP0 has to send out can then be reduced.

4.2 Out-of-Sequence Problem

The architecture shown in Fig. 5 may route packets with the same line number to different output lines. For example, n packets from an input line have been sent to out1. But the (n+1)th packet may be switched to out2 because reconfiguration took place to balance traffic among the output lines. This may cause an out-of-sequence problem after the packets are routed through a packet switching fabric if the (n+1)th packet is routed to the output port before the nth packet is. To avoid the problem, one rule must be followed : the input lines carrying services with high bit rate information, such as video signals, will not be switched to another output line during the service period. For the low bit rate services, such as voice - 64 Kb/s, even if the packets are dispatched onto two output lines, they are separated by more than several hundred packet intervals. Hence, they won't cause an out-of-sequence problem when they arrive at the receive end. The performance of the N:M packet multiplexer will definitely degrade because of the constraint, but is still greater than that of M multiplexers with each one having N/M inputs and one output.

When a new call with high bit rate information is requested on an input line, the input line may be assigned to another group. This causes the switch connections to change. But, if a input line remains in the same group, then no switch reconfiguration may be necessary, unless the priority between the input lines in the group needs to change. The controller has a table containing information about the input lines' and output lines' traffic statistics. To assign the input line to the group having the lowest traffic statistics may not be the best choice in reality. For instance, if an input line in group g is assigned to another group g' for traffic balancing purpose, the lines in both g and g' groups are affected because of the reconfiguration of the switching network. As explained above, the affected lines cannot send out packets during network reconfiguration to prevent packets from being corrupted. If the input line remains in group g, then no reconfiguration will occur and hence there will be no interruption in sending packets.

Therefore, when a new call with high bit rate information occurs, there is a tradeoff between keeping the traffic even and keeping the configuration as static as possible. Note that as a call is terminated, it doesn't cause a network change. To insure fairness, it is necessary to occasionally reshuffle the line priorities in the group. But, the line is not assigned to a different group to prevent the out-of-sequence problem. When network connections are changed because of a new call setup, it is a good opportunity to change the service order of the input lines.

4.3 Multiple ADROP0s

An alternative architecture increases the number of the ADROP0s from one to M and the size of the switching network from (N+M)x(N+M) to (N+2M-1)x(N+2M-1). Each group would have it's own ADROP0 chip sending out empty frames. The regroupme scheme discussed above can be employed in this case too. There are two advantages for having M ADROP0s instead of one. The first is

that when network reconfiguration occurs, the only input lines affected are those having a connection changed. The second is that the failure of one ADROP0 won't disable the system.

5. Packet Demultiplexer

The function of the Packet Demultiplexer is to assign an incoming packet from a central office to an individual subscriber access channel. The distributed architecture shown in Fig. 7 follows the one in [6] except that it uses ADROP chips to do the packet filtering function.

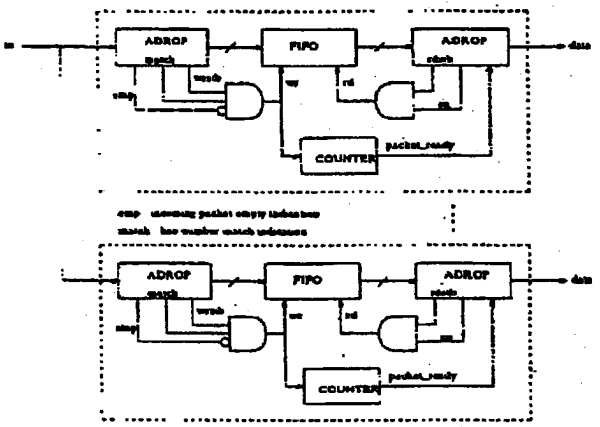

Fig. 7. Packet demultiplexer with distributed control

An incoming frame is simultaneously received by each ADROP in the input port. The packet occupancy and line identification number in the packet header are examined by the ADROPs. If the packet is not empty and the line number is matched, the packet will be written to the FIFO and then read out by the ADROP at the output port. Otherwise, the packet is simply discarded. A counter in the figure is used to count the number of bytes written into the FIFO and generates a "packet_ready" signal when a full packet is stored in the FIFO. This signal will inform the ADROP at the output port to start reading the packet in the FIFO. The ADROP will assert the "en" signal during the reading of the entire frame interval. The ADROP functions at the input and output ports in the Packet Demux can be performed by a single ADROP chip.

6. Specifications and functions of the ADROP Chip

The ADROP chip is an essential component to implement the architectures discussed above. This section summarizes the functions necessary for the ADROP chip and presents a primitive block diagram of the chip. The detailed circuit design of the chip cannot be attempted until a complete packet structure is determined. Therefore, the chip at present will only contain some basic functions that are independent of the packet structure. For prototyping the architectures in the early stages, those functions related to the detailed packet structure can be realized with external circuits, such as programmable array logic (PAL).

The primary functions of the chip include serial to parallel and parallel to serial conversions, frame byte detection, control byte insertion, packet occupancy detection, line (or channel) identification matching, and the ability to overwrite serial data. In many cases, the packet header has to be examined first and associated signals generated to control peripheral circuits. For example, in the Packet Disassembler and Packet Demultiplexer whether or not a packet is to be written into the FIFO depends on the "emp" and "match" signals, which are generated corresponding to the packet occupancy and line (or channel) identification number. Therefore, it is necessary to access the packet header and generate proper signals before the packet flows out the ADROP chip. Since the detailed packet structure is not yet determined, the circuits for detecting the packet occupancy and line (or channel) ID will not be built inside the chip. Instead, a K-byte delay circuit will be added in the chip as shown in Fig. 8 to allow users access to the packet header K bytes before the packet flows out of the chip.

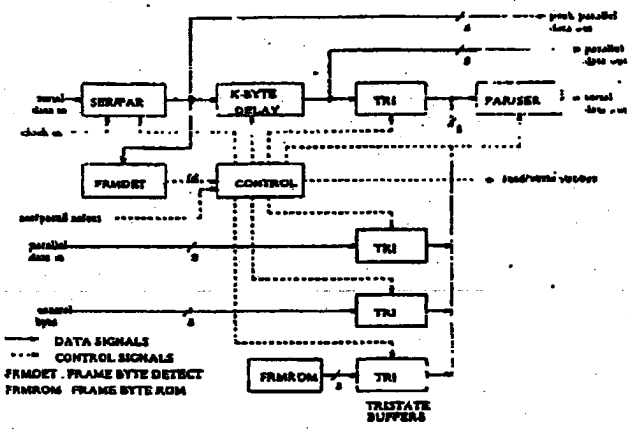

Fig. 8. Block diagram of the ADROP chip

Except for the K-byte delay circuit, all other blocks shown in the figure have been built in the Framer chip.[10] This full custom 2 micron CMOS VLSI chip was fabricated, tested and fully functional up to 100 Mb/s. Based on the first chip's test results, the next generation of chip has been modified for increased speed. Timing simulation results showed that the revised chip should operate at 150 Mb/s. Because the information to be examined in the packet header may be the last byte of the header, the value of the K should be larger than the header length plus the time intervals needed to generate proper control signals, usually less than one byte time. The new chip will provide two sets of parallel data outputs associated with separate sets of read/write strobes. They are separated by K bytes interval. Using the ability to "peek" at the serial data input from the peek parallel data output before it goes out of the chip, users can overwrite particular bytes in the serial bit stream by inserting the updated bytes at the parallel data input port at the proper time.

7. Conclusions

In this paper we have examined the design architectures of two multiplexers required in the DTDM network. We first described a simple modularized design architecture for a Packet Assembler, based on a daisy-chained structure. Then we described a novel multiple input-multiple-output statistical multiplexer design that incorporates a cross point switch network. Finally, we discussed the design of a common crucial component, the ADROP chip, in more detail.

8. Acknowledgement

The author wishes to express his gratitude to S. H. Lee for his invaluable discussions and stimulation. He would like to thank M. W. Beckner for his comments and suggestion about having more than one ADROP0 in the N:M packet multiplexer to improve system reliability. He also would like to thank L. S. Smoot for his support and comments.

REFERENCES

[1] L. T. Wu, S. H. Lee and T. T. Lee, "Dynamic TDM: A Packet Approach to Broadband Networking," in Proc. International Conference on Communications, Seattle Washington, June 1987.

[2] W. W. Chu, "A Study of Asynchronous Time Division Multiplexing for Time Sharing Computer Systems," in Proc. AFIPS, vol. 35, pp. 669-678, 1969.

[3] A. Thomas, J. P. Coudreuse and M. Servel, "Asynchronous Time Division Techniques: An Experimental Packet Network Integrating Video Communication, " in Proc. International Switching Symposium, May 1984.

[4] J. Hui and E. Arthurs, "A Broadband Packet Switch for Multirate Services," in Proc. International Conference on Communications, Seattle Washington, June 1987.

[6] L. R. Linnell, "A Wide Band Local Access System Using Emerging Technology Components," IEEE Journal on Selected Areas in Communications, Vol. SAC-4, No. 4, pp. 612-618, July 1986.

[7] M. W. Beckner, T. T. Lee and S. E. Minzer, "A Protocol and Prototype for Broadband Subscriber Access to ISDNs," in Proc. International Switching Symposium, March 1987.

[8] J. Y. Cochennec, P. Adam and T. Doudoin, "Asynchronous Time-Division Networks : Terminal Synchronization for Video and Sound Signals," in Proc. Globecom, New Orleans, Dec. 1985.

[9] G. Hayward, A. Gottlieb, D. Boyer, and J. Berthold, "High Speed 16x16 CMOS Crosspoint Switch," Electronics Letters 1985, 21, pp. 923-924.

[10] D. Boettle and M. Klein, "High Speed (140 Mbit/s) Switching Techniques for Broadband Communications," in IEEE Proc. 1986 International Zurich Seminar on Digital Communications, March 1986.

[11] H. J. Chao, T. J. Robe, and L. S. Smoot, "A CMOS VLSI Framer Chip for a Broadband ISDN Local Access System," in IEEE Digest of Technical Papers of 1987 Symposium on VLSI Circuits, Karuizawa Japan, May 1987.

EXHIBIT 7

A 140 Mbit/s CMOS LSI Framer Chip for a Broad-Band ISDN Local Access System

H. Jonathan Chao
Thomas J. Robe
Lanny S. Smoot

Reprinted from
IEEE JOURNAL OF SOLID-STATE CIRCUITS
Vol. 23, No. 1, February 1988

IEEE JOURNAL OF SOLID-STATE CIRCUITS, VOL. 23, NO. 1, FEBRUARY 1988

A 140 Mbit/s CMOS LSI Framer Chip for a Broad-Band ISDN Local Access System

H. JONATHAN CHAO, MEMBER, IEEE, THOMAS J. ROBE, MEMBER, IEEE, AND LANNY S. SMOOT, MEMBER, IEEE

Abstract—A CMOS LSI framer chip, which provides a SONET-like time-division multiplexed (TDM) frame structure, and which has been implemented in a production 2-μm technology, is described. Current samples of the chip have been tested functional to 160 Mbit/s. The primary attributes presented include the circuit features employed to achieve high-bit-rate operation, the level translation circuits which afford direct interfacing to ECL level clock and data signals, and the functional capabilities which lead to broad application of the chip in communications networks. Several examples of its use in a prototype broad-band local access network are also described. The chip operates from a single 5-V supply, dissipates approximately 420 mW, and is packaged in a 68-pin leadless ceramic chip carrier (LCCC) package.

I. Introduction

THE EMERGING Broad-band Integrated Services Digital Network (BISDN) will satisfy subscriber's present and future needs for a wide range of telecommunications bit rates and holding times. A BISDN subscriber access prototype [1], [2] being explored at Bell Communications Research includes high-speed circuit switching, time-division multiplexing, packet multiplexing, and packet switching [3]. A double star architecture was proposed for the broad-band communication system. Single-mode fiber feeders form a first-level star connection from the broad-band central office (CO) to a remote electronics (RE), where upstream traffic is multiplexed and downstream traffic is demultiplexed and circuit switched to individual subscribers. Distribution and drop single-mode fibers form a second-level star to subscribers, as shown in Fig. 1.

Each direction of a subscriber drop may contain up to four identical channels. One of the channels must be used to provide the subscriber with packet access since message-based signaling over the packet channel is used for communication with the broad-band CO. The remaining channels may be used for either packet or circuit traffic (e.g., full-motion video). Each of the basic channels is referred to as a WB-1 channel, and the resulting structure when four such channels are time-division multiplexed (TDM) on the local loop is a WB-4 channel. The ultimate bit rate for the basic channel, WB-1, has not been fixed but it is suggested that the bit rate should be in the range of 130–150 Mbit/s. It is anticipated that channel bit rates in this range will allow for cost-effective coding of extended quality digital video signals and cost-effective multiplexing and transmission of four such channels.

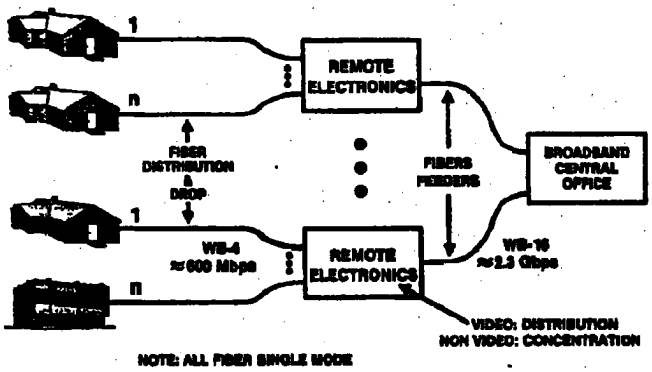

Fig. 1. Broad-band ISDN local access architecture.

In order to identify data streams in the network, and to locate individual bytes within these streams, a synchronous SONET-like [4] frame format for the basic channel is adopted for use in the prototype as shown in Fig. 2. It consists of eight subframes, in each of which there are eight blocks. Each block contains 34 bytes with the first byte being a frame byte, span (tributary) identification byte, or control byte. This results in the information payload being 33/34, or 97 percent. The frame repetition rate of 125 μs facilitates the definition of 64-kbit/s channels. The bit rate is (8 subframes × 8 blocks × 34 bytes × 8 bits)/125 μs, or 139.264 Mbit/s.

A full-custom 2-μm CMOS LSI framer chip which provides access to this framed channel has been designed, fabricated, and tested. The chip has application at the subscriber interface to the network, in the TDM transmission system linking the various system nodes, and in the packet processor at the CO (Fig. 1). The basic functions of the framer chip are:

1) *framing*—converting 8-bit parallel data into a WB-1 serial bit stream with the frame structure shown in Fig. 2;

Manuscript received July 22, 1987; revised October 1, 1987.
The authors are with Bell Communications Research Inc., Morristown, NJ 07960.
IEEE Log Number 8718222.

0018-9200/88/0200-0133$01.00 ©1988 IEEE

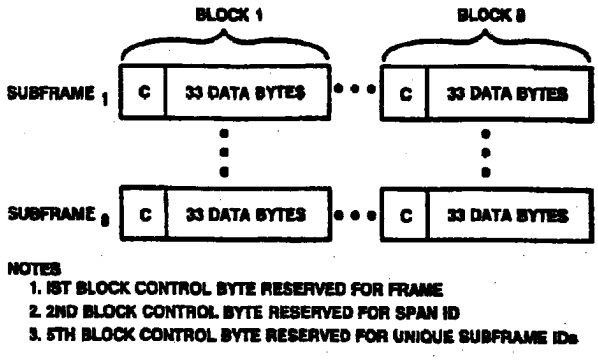

Fig. 2. Synchronous broad-band transmission format.

2) *deframing*—detecting the frame-byte location within an incoming WB-1 bit stream and converting the input serial data into a parallel output format;
3) *control-byte add/drop*—inserting and detecting control bytes in certain applications, such as span (tributary) identification in a TDM transmission system; and
4) *data-byte add/drop*—reading data bytes present in the input serial bit stream and, if appropriate, overwriting them at the serial data output.

Because the high-speed CMOS framer chip was not available during construction of the BISDN prototype, an MSI piece part framer board was substituted in its place. The chips, which are now available, can replace these boards and result in more than an order of magnitude savings in space and power for this function. The complete architecture of the chip at the gate level was designed on a commercial workstation. The layout was created using MULGA [5], [6], an in-house symbolic design tool, with which designers construct the transistor-level circuits, interconnect them on a virtual grid, and then compact them based on a specific set of 2-μm design rules. The compacted circuit layouts were then extracted and simulated using SPICE, a timing simulator, and a logic simulator.

This paper addresses the applications of the generic framer chip in a broad-band network system in Section II. The chip functionality is described in detail in Section III. In Section IV, we present some examples of high-speed digital and analog circuit designs in the chip, the frame alignment procedure, and handshake signal generation. Finally, we show the performance results and conclude the paper in Sections V and VI, respectively.

II. Framer-Chip Applications

A. User Network Interface

Fig. 3 shows how the framer chip can be applied in a User–Network Interface (UNI). The UNI is the means by which subscriber terminal equipment, i.e., telephones, data terminal equipment, video codecs, etc., access the WB-1 serial bit stream. In this application a byte-wide first-in/first-out (FIFO) buffer and a framer chip provide the interface between transmit or receive equipment and the network. At the transmit end, the chip generates strobes to read data and control bytes from the FIFO to its parallel data input (pdi) port and produces a properly framed data stream at its serial data output (sdoe) port. At the receive end, the chip recognizes a framed data stream applied to its serial data input (sdie) port and presents the data bytes on its parallel data output (pdo) port to the FIFO along with the appropriate WRITE strobes.

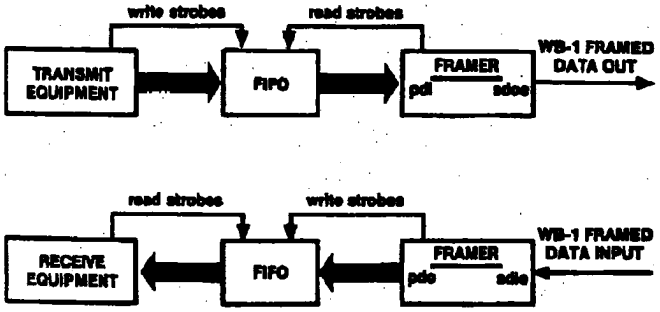

Fig. 3. Framer chip used in UNI.

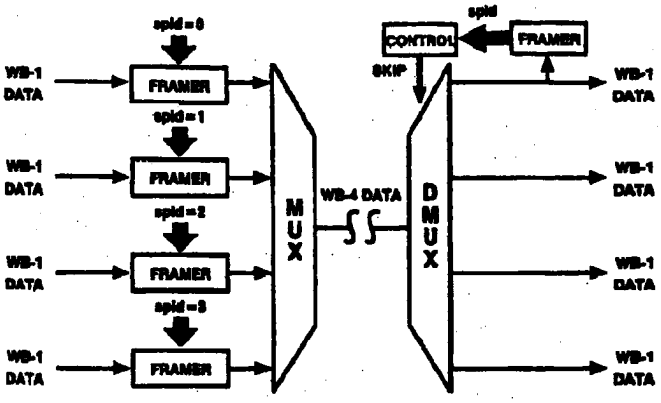

Fig. 4. Framer chip used in MUX/DEMUX system.

B. TDM Tributary Identification

Since the chip allows for user insertion and detection of specific control bytes, it can serve in a time-division multiplex/demultiplex (MUX/DEMUX) system by tagging MUX input tributaries and identifying DEMUX output tributaries, as shown in Fig. 4. A predetermined span (tributary) identification (spid) byte is inserted at each input tributary and four WB-1 channels are then multiplexed to generate a higher speed data link (WB-4). At the DEMUX subsystem, an incoming WB-4 bit stream is demultiplexed into four tributaries. One of the tributaries is connected to a framer chip, which passes the spid to a controller. The controller can then determine whether the bit stream is on the correct tributary by examining the spid. If not, a SKIP pulse is generated and the DEMUX will reassign the order of the bit stream. Hunting for the right spid will continue until the proper WB-1 channel is physically placed.

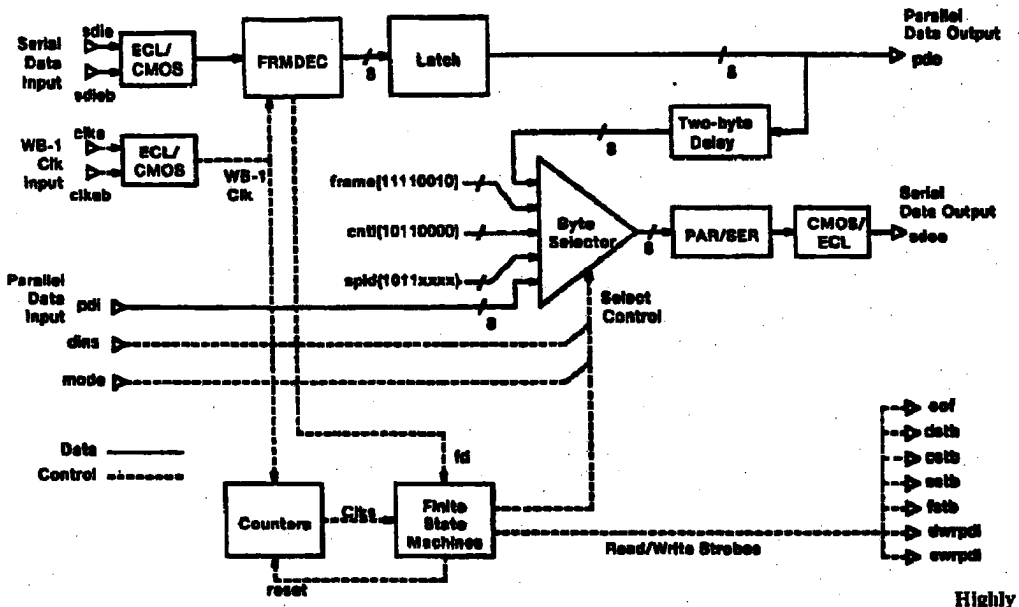

Fig. 5. Framer-chip block diagram.

C. High-Speed Packet Processing

The BISDN CO packet processor transforms incoming packet headers (line number and logical channel number) into the routing information required by the self-routing packet switch used in the broad-band prototype. It also counts packets and enforces bandwidth limitations on logical channels. In this application the framer chip is used in a manner similar to the UNI. The framer receive function converts the serial bit stream to 8-bit parallel form and separates the packet headers from the packet body. The headers are routed through a header processor while the body is routed to a parallel FIFO. The framer transmit function then reassembles the body with the altered header, inserts frame and control bytes, and serializes the result onto the WB-1 bit stream. A single framer chip provides both receive and transmit functions.

III. Framer-Chip Architecture

A. Data Flow

The framer functional block diagram of Fig. 5 has three input-to-output data paths. At the top of the figure is the serial-data-input to parallel-data-output path, which requires frame synchronized clocking of the latch in order to correctly group the pdo bytes. This path is always enabled. The other two paths are from serial-data input to serial-data output and from parallel-data input to serial-data output. One of these two paths is selected via the byte selector by asserting the "dins" pin high (parallel to serial) or low (serial to serial). Simultaneous operation of the serial-to-parallel and parallel-to-serial paths is possible, allowing a single chip to function as both a data transmitter and data receiver. Moreover, the two-byte delay provides the ability to examine data bytes at the serial input and to overwrite them from the pdi input, when operating in the serial-pass-through mode. This add/drop capability is important in packet transmission applications where the framer chip can function as an essential component of a packet assembler and a packet multiplexer [7]. In addition to data-path selection, the byte selector provides the means by which frame bytes, span-identification bytes, and control bytes are inserted into the sdoe stream to give the frame structure of Fig. 2. Byte insertion is controlled by synchronization signals from a finite-state machine. The frame and control bytes and the four most significant bits of spid are fixed by internal chip wiring, with the four least-significant bits of spid set at external pin connections. In this way, up to 16 different spid's are possible. A final option available via the byte selector is the ability to overwrite the fixed control bytes with bytes from the pdi port by asserting the "mode" pin high. For example, these bytes can be used to provide an out-of-band signaling capability for network maintenance. The frame and spid bytes cannot be overwritten from the pdi port.

B. Finite-State Machines and High-Speed Counters

All timing control signals, READ/WRITE strobes, and the out of frame *oof* signal are generated by five finite-state machines, which are driven by a set of three counters that provide a byte clock, block clock, and subframe clock. When a framed bit stream is applied at the serial input, the frame-byte detector (FRMDEC) and one of the finite-state machines implement a search algorithm to align the clocks and strobes to the input pattern and to declare an in-frame condition, i.e., *oof* low. Upon detection of a frame byte in FRMDEC, a 1-bit-wide signal *fd* is sent to a state machine which, in turn, resets the counters to properly define all clock boundaries. Three correctly spaced consecutive occurrences of *fd* are required to declare an in-frame condition. Successful alignment results in parallel output bytes which are properly grouped and strobes (*dstb*, *fstb*,

etc.) correctly timed to read them. In some applications there are no serial input data on which to frame. This is true, for example, when the chip is used in the transmit mode only. In this condition the counters free-run; i.e., they are not reset, but generate the same set of frame clocks necessary to give a properly framed pattern at the serial output port. The write strobes *dwrpdi* and *cwrpdi* are also timed to accurately write data in at the pdi port.

The blocks labeled ECL/CMOS and CMOS/ECL are on-chip level translators which allow for direct interface to ECL-level serial bit streams and clock signals. They are described in detail in Section IV. The PAR/SER block is a parallel-to-serial converter which results in the serial bit stream at the sdoe.

IV. Chip Implementation

This section describes the critical circuits and concepts used in the chip. Since the speed performance goal for the chip is 140 Mbit/s, all circuits related to the serial bit stream must be optimized for speed. The chip operates synchronously; data and signal flow are controlled by clocks. Pipeline techniques are used to reduce the propagation delay between registers (flip-flops). As is well known, the maximum clock rate for a synchronous system is determined by the sum of 1) the setup time of a flip-flop, 2) the delay time of a flip-flop from clock to output, and 3) the propagation delay time of any combinational circuit between the flip-flops. The first two terms are fixed for a chosen flip-flop. In order to maximize the chip speed, we limited the number of stages between the flip-flops to no more than three.

A. D-Type Flip-Flops (DFF's)

Two types of *D*-type flip-flops (DFF's), a static DFF and a dynamic DFF, are used and are shown in Fig. 6(a) and (b), respectively. Because both have master and slave latches that store data, they are equivalent to edge-triggered DFF's. The static DFF's are used in circuits which are clocked at the 17.4-MHz byte rate or less, and the faster dynamic DFF's are used in a counter, two shift registers, and several finite-state machines which are clocked at the WB-1 rate (139.264 MHz). The simpler dynamic DFF, which relys on internal device capacitance to store its data, operates approximately 2 ns faster than the static DFF. Because it is clocked continuously at the WB-1 rate, it can easily store data on its small device capacitances. Use of this fast flip-flop in the speed-limiting paths of the chip contributed significantly to achieving the 140-Mbit/s design goal.

Fig. 6(a) and (b) shows that the flip-flops are clocked with two clock phases, and therefore, it is important to consider the effect of clock skew. To the extent that the two clock phases do not change state at the same time, there will be a corresponding increase in the hold time, setup time, or propagation delay of the flip-flops. This can produce an adverse effect in the critical speed-limiting

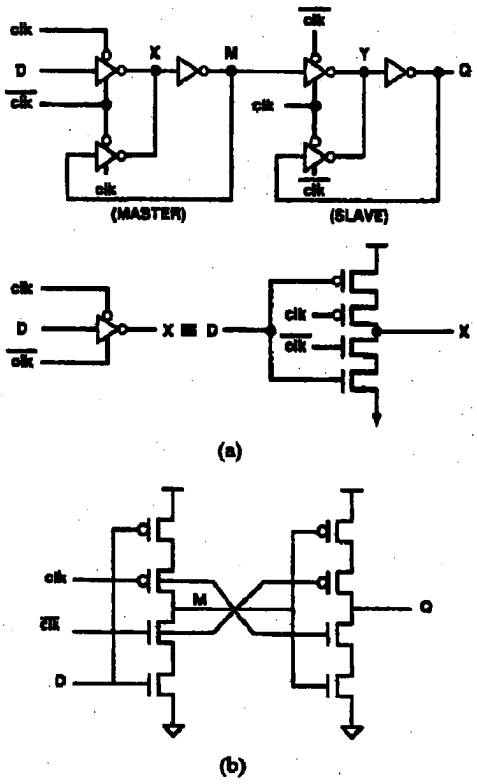

Fig. 6. (a) Static DFF schematic diagram. (b) Dynamic DFF schematic diagram.

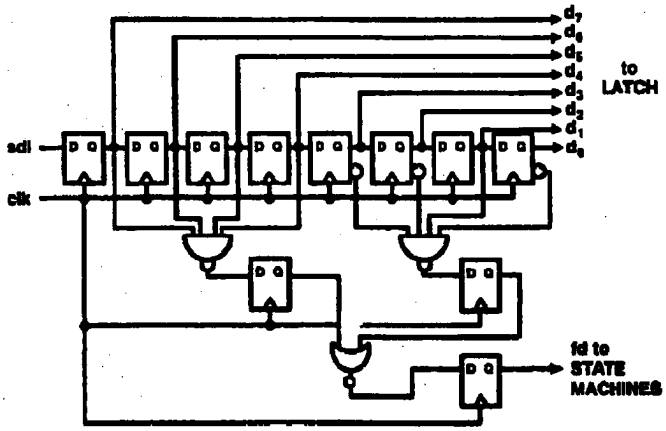

Fig. 7. Logic diagram of frame-byte detector (FRMDEC).

circuits. In order to prevent this, care was given to the distribution of the clock phases by balancing the clock drive circuitry, the distribution wiring, and the clock loads.

B. High-Speed Logic Circuits

Two high-speed circuits, FRMDEC and a divide-by-8 counter, are presented here. Fig. 7 shows the logic diagram of the FRMDEC, in which the serial data input is clocked into an 8-bit-wide shift register and then byte-wide data, d_7 to d_0, appears at the output. The shifted bits are compared with a predetermined frame byte, "11110010," which was selected because no part of this byte can combine with adjacent bits to form a false frame byte. A bit-wide pulse *fd* is generated and routed to the input of a state machine whenever the pattern is detected. Because

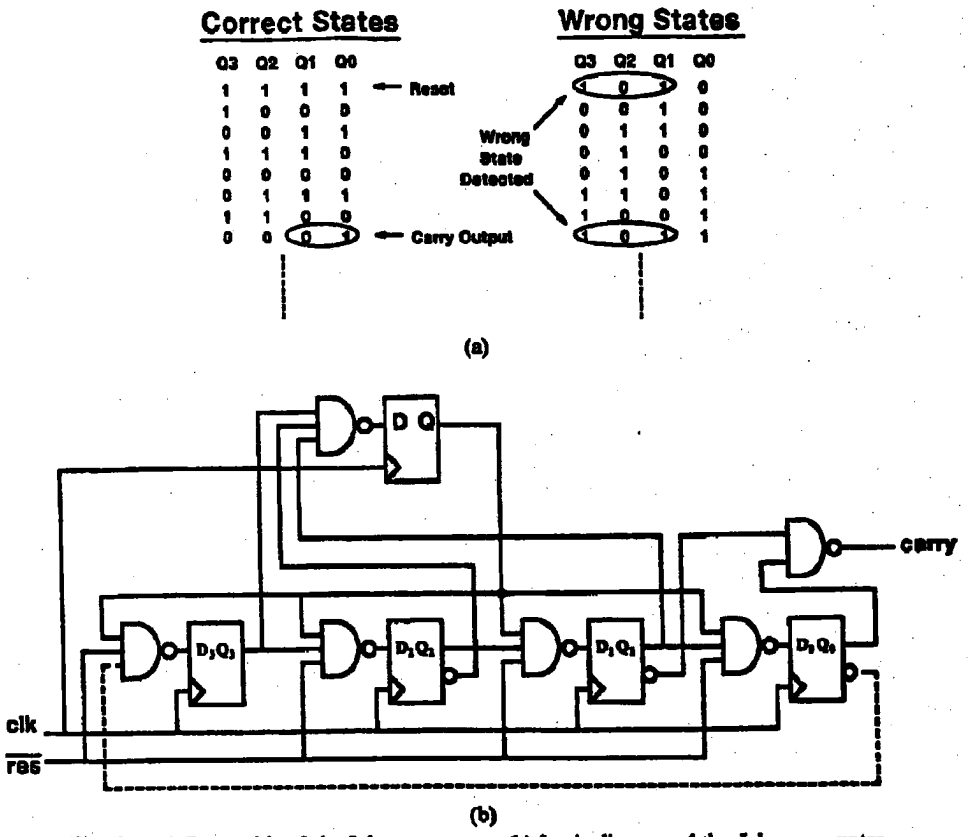

Fig. 8. (a) State table of the Johnson counter. (b) Logic diagram of the Johnson counter.

the pattern resides in the shift register for only one clock cycle, and because the path traversed by *fd* includes the four-input NAND gate, the two-input NOR gate, and the state-machine input logic, it was necessary to pipeline this function by inserting the two DFF stages shown between the NAND gates and the NOR gate.

Three counters are employed to generate byte clock (divide-by-8), block clock (divide-by-34), and subframe clock (divide-by-8), respectively. The first counter generating byte clock must operate at high speed, 140 MHz, and was built with a Johnson counter employing the fast dynamic DFF's. The other two counters, which operate at less than 20 MHz, are conventional synchronous toggle counters and use the static DFF. Since the Johnson counter has four state variables to perform the divide-by-8 function, it has 16 possible states. They are divided into two groups, with eight states in each (Fig. 8(a)). When the counter is reset, the state variables Q_0–Q_3 are "1111." A carry pulse is generated when the state variables are "0001." Since the two least significant bits, Q_0 and Q_1, have the value "01" in only one of the 16 states, we used these two state variables to generate the divide-by-8 output. The dashed line shows the critical path of the counter, which contains a three-input NAND gate between two DFF's plus the DFF $\overline{Q}$ output inverter. In order to prevent the counter from looping in the wrong states, it is equipped with a *wrong state detector* which resets the counter to a correct all ONE's state if a "101x" state is detected. It is shown in the upper portion of Fig. 8(b). Selecting the "101x" state for reset helps to evenly distribute the flip-flop loading so that a single flip-flop is not excessively loaded and, consequently, more speed limited. This self-correcting circuit does not affect the counter speed because it is not within the critical path.

C. Frame Alignment Algorithm

Frame alignment discussed here really means subframe alignment. There are many different frame alignment techniques. We present a frame alignment algorithm used in the chip, which is simple and easy to implement with a finite-state machine. Since the information bits in the subframe may be random, it is likely that they will imitate a frame byte. Moreover, once the subframe is aligned, transmission errors can cause occasional corruption of a frame byte, which may be ignored. However, a more severe disturbance, corrupting many frame bytes, requires system realignment. Fig. 9 shows the state-transition diagram of the alignment procedure. When the system is initialized, a state machine in the chip is reset to a SEARCH state. Whenever a frame-byte pattern is detected in the incoming serial bit stream, an *fd* pulse is generated in the FRMDEC indicating FOUND. The state machine moves to the "CONFIRM 1" state and resets the counters which define byte boundaries. Once a frame byte is found, the state machine reexamines the bit stream one subframe later (272 bytes). Then, if a frame-byte pattern is not found, the state machine returns to the SEARCH state. In this way, the chip will not declare itself *in-frame* until frame bytes are found at three consecutive subframe intervals. At that

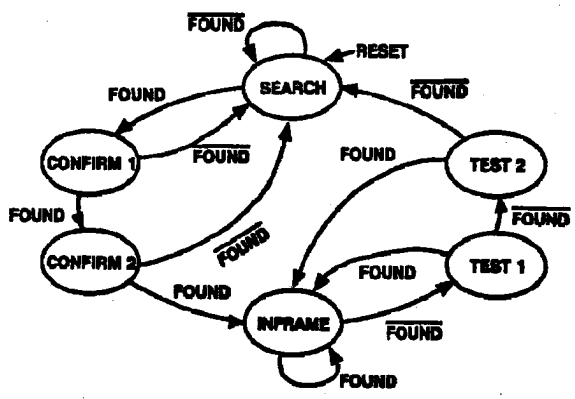

Fig. 9. State-transition diagram to implement reframing algorithm.

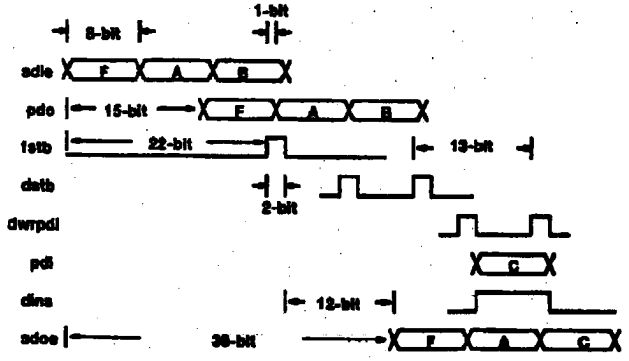

Fig. 10. Timing diagram describing the overwrite operation.

moment the out-of-frame *oof* signal becomes false (low). Similarly, the state machine will not declare *out-of-frame* (return to SEARCH state) unless frame bytes are missed for three consecutive subframe intervals. The *oof* signal is then asserted high.

The average time taken to align the subframe depends on the subframe size, subframe rate, and the frame-pattern length; it is given as [8]

$$T_F = \frac{y}{\text{subframe rate}}\left(\frac{N+1-Bx8}{2^F-1}+1\right)+2(\text{subframe time})$$

where y is the fraction of the subframe time from the search point to the frame byte, N is the number of bits within one subframe ($N=2176$), B is the number of transmission overhead bytes in one subframe ($B=8$), and F is the frame-pattern length ($F=8$). The subframe rate is 64-kbit subframes/s. By substituting the above values, the maximum average reframing time ($y=1$), $T_{F,\max}$, is then equal to 11 subframe periods, or 171.875 μs.

D. READ/WRITE Strobe Timing

As described in Section III, two of the finite-state machines generate READ and WRITE strobes which are used externally to read bytes from the pdo port or to write bytes at the pdi port. The READ strobes, *dstb*, *cstb*, *fstb*, and *sstb*, are 2-bit-wide pulses, only one of which is asserted for each byte at the pdo port. In this way the parallel output bytes are identified as data bytes, control bytes, frame bytes, or span-identification bytes depending on the strobe asserted. Since 33 of every 34 bytes in a block are data bytes, *dstb* occurs 264 times each subframe (272 bytes) whereas *fstb* and *sstb* occur only once each subframe. Taken together the combined READ-strobe rate is one-eighth of the WB-1 rate. Two WRITE strobes are generated, *dwrpdi* and *cwrpdi*. These are used for writing data or control bytes into the pdi port. However, they will do so only if the "dins" or "mode" pins are asserted high.

A timing diagram illustrating the data-byte overwrite process is given in Fig. 10. The sdie stream contains a frame byte F followed by the data bytes A and B. These bytes appear at pdo 15-bit periods later and can be latched externally by the strobe signals *fstb* and *dstb*. In order to overwrite data byte B with data byte C, the user can read byte B with the rising edge of the second *dstb* pulse, make the decision to perform the overwrite, and assert the data insert pin "dins" high prior to the rising edge of the second data-WRITE signal *dwrpdi*. The chip signal *dwrpdi* latches the new data byte C in the output parallel-to-serial shift register. Thirteen bit periods are available, from the rising edge of *dstb* to the rising edge of *dwrpdi*, within which to read the data byte and to activate the overwrite procedure.

E. ECL–CMOS Translators

The framer clock input (clke and clkeb), serial data input (sdie and sdieb), and serial data output (sdoe) are at 140 MHz and 140 Mbit/s, respectively, and are supplied at ECL interface levels. Because the chip also interfaces more than 30 TTL level I/O's, it operates from a +5-V supply. Consequently, ECL-to-CMOS and CMOS-to-ECL level translation circuits are needed at its 140-Mbit/s serial inputs and outputs. Although it was thought desirable and feasible to build the translation circuits completely on the framer chip itself, the negative ECL voltages cannot be applied directly to its terminals because this would result in potentials greater than 5 V across the chip. To accommodate this, external low-voltage zener diodes are combined with on-chip active components to implement the level-shift functions.

The clock input ECL-to-CMOS translator circuit is shown in Fig. 11(a). It is simply two separate differential-to-single-ended converters operated in a quasi-linear fashion to provide CMOS clk and clkb at the two outputs. The differential ECL inputs of each converter are coupled through external 3.3-V zeners so that the levels at the NMOS transistor gates are always positive and bias the NMOS transistors for high bandwidth. Bias current for the zeners is furnished by chip PMOS current source (CS) transistors which employ a single external current set resistor (RBI). This resistor could have been implemented on-chip, however, in order to evaluate resistor level shifting as well as zener level shifting the external construction was used. The input translators can be driven by a single-ended input against a reference voltage of negative 1.3 V, but operation in this manner does not provide sufficient drive for the clock at 140 MHz. Therefore, in the BISDN

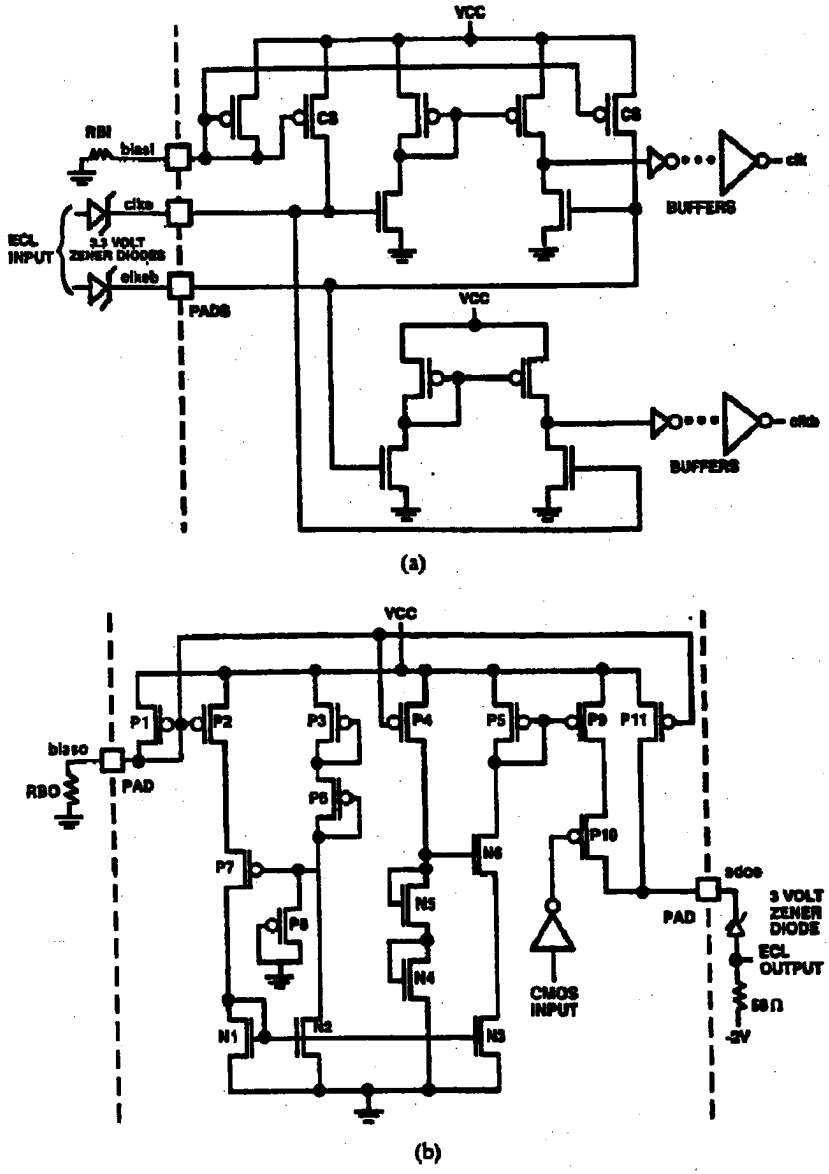

Fig. 11. (a) ECL-to-CMOS translator schematic. (b) CMOS-to-ECL translator schematic.

prototype applications described, differential ECL input is required. The translated CMOS level outputs directly drive scaled multiple inverter buffers for chip distribution of two phase clocks clk and clkb to all clocked loads. The data-input translator uses a single identical converter because data are distributed single phase.

Correct phasing of clock and data is established external to the framer chip and must therefore be maintained through the level-translation process. Accordingly, the translators and multiple inverter buffers for clock and data are identical. Even so, the clock load is much larger than the data load and requires that the final buffer size be sufficiently large such that its delay is not strongly load dependent.

The data output CMOS-to-ECL translator is shown in Fig. 11(b). When in the high output state, its output is a current source of approximately 20 mA through a 3-V zener to a standard 50-Ω ECL load. The zener is used in order to assure that the framer output terminal is always positive. In the low output state, a small current is sourced by transistor P_{11} to bias the zener.

The high-state current, sourced by transistors P_9 and P_{10}, is controlled by the current mirror network consisting of transistors P_1–P_8 and N_1–N_6. It originates as a low current in P_1 and is multiplied by approximately 20X to P_9 in two stages. Accurate mirroring from P_1 to P_2 and from N_1 to N_3 requires that the two P transistor V_{DS}'s closely match and the two N transistor V_{DS}'s closely match. The networks P_3, P_6, P_7 and N_4, N_5, N_6 serve to provide this voltage match. Transistor P_8 is a small device, whose function is to draw a small current through P_3 and P_6 to assure turn on of P_7.

Transistor P_9 has its gate continuously biased for full output current and source current control is provided by switching P_{10} on or off. This series operation at high current is not desirable because of the increased voltage drop across the output current source transistors. However, if a single transistor output were employed, it would

be necessary to switch the gate voltage at P_9 via P_5. Operation in this manner would result in decreased speed because, although P_5 can turn on rapidly, it cannot turn off fast enough for operation at 140 Mbit/s.

The translator circuit as shown provides ECL 10KH compatibility in the laboratory environment used to evaluate the framer chip, but would require improved regulation of the reference current in transistor P_1 for full 10KH compatibility in a field environment.

V. Measured Performance

A printed-circuit evaluation board was constructed to test the functionality and speed of the framer chip. Because we do not have a word generator which can provide a 2176-bit 140-Mbit/s serial bit stream, two framer chips are used so that the first (transmitter) can provide the framed serial data stream needed to test the second (receiver).

Complete chip functionality, including ECL-to-CMOS and CMOS-to-ECL translation, frame-byte detection, correct counter reset, serial-to-parallel conversion, and parallel-to-serial conversion can be verified by monitoring the parallel-data output, the READ/WRITE strobes, and serial-data output of the receiver framer chip. A fixed data pattern, 10010110 (96_H), is asserted at the pdi of the transmit framer chip. Its ECL sdoc is converted to a differential signal by an external ECL gate and is then fed into the sdie of the receive framer chip. The receive-chip pdo and its READ/WRITE strobes were monitored with a logic analyzer. Its CMOS-level serial-data output (sdoc) was observed with an oscilloscope. Fig. 12 shows the correct timing relationship between read strobes, *fstb* and *dstb*, and the sdoc at 160 Mbit/s. The frame byte, 11110010 ($F2_H$), appears at sdoc 12 bit periods after the falling edge of *fstb* with its least-significant-bit leading. Preceding and following the frame byte are the fixed data patterns 10010110.

The maximum speed of the parallel-to-serial conversion was verified to 175 Mbit/s by examining the transmit framer-chip sdoc. Above 175 Mbit/s, the Johnson counter, which controls the loading of bytes into the output parallel-to-serial shift register, failed to divide by eight properly.

The 160-Mbit/s serial-data-input limit is caused by a critical timing path through a DFF and a gate in the state machine which resets the counters, and a NAND gate and a DFF in the Johnson counter. A test circuit modeling this known critical path was included on the chip and is shown in Fig. 13. The DFF in the figure models both the delay time of the state-machine DFF and the setup time of the counter DFF. The gates model the intervening stage delay and loading. The odd number of gates in the path makes testing a simple matter of feeding back and measuring the maximum toggle rate of the test circuit. As expected, this limit was also approximately 160 MHz. This test circuit, as well as buffers for viewing internal test points, is enabled by an external "test enable" pin which is not asserted during normal chip operation in order to reduce noise and power dissipation.

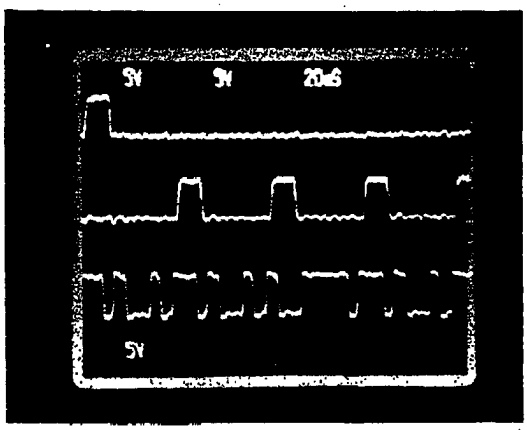

Fig. 12. Measured timing of *fstb*, *dstb*, and sdoc at 160 Mbit/s.

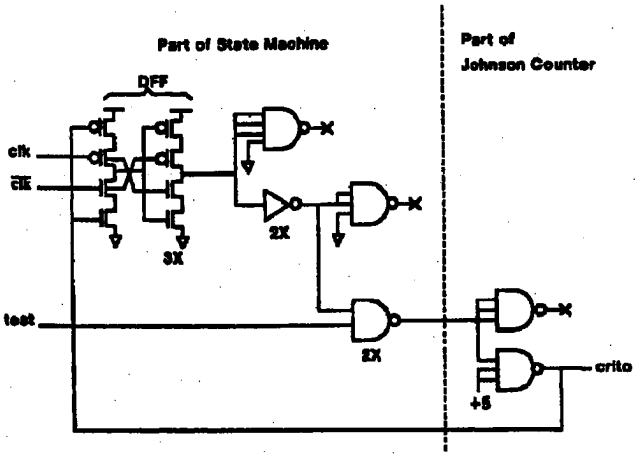

Fig. 13. Test circuit modeling the speed-limiting critical path.

VI. Conclusions

A 2-μm CMOS LSI framer chip providing receive and transmit access to a 140-Mbit/s serial channel has been designed, fabricated, and tested. In addition to serial-to-parallel and parallel-to-serial conversion, it formats the serial output bit stream in a SONET-like frame including frame-pattern and span-identification overhead bytes for easy data recovery. Functioning as a data receiver, the chip detects frame bytes and produces properly grouped 8-bit data at its parallel output port. An add/drop function affords the ability to examine the received data and to overwrite it at the serial output when operating in the serial-pass-through mode. The design goals for this chip include clock processing and distribution at 140 MHz, CMOS-to-ECL translation at 140 Mbit/s, frame-byte detection at 140 Mbit/s, and counter operation at 140 MHz. The current design is fully functional to 160 Mbit/s, dissipates approximately 420 mW, and is packaged in a 68-pin leadless ceramic chip carrier (LCCC) package. Fig. 14 shows a photomicrograph of the 5×5-mm^2 framer chip. A V_{dd} and V_{ss} pad is placed at each of the four corners.

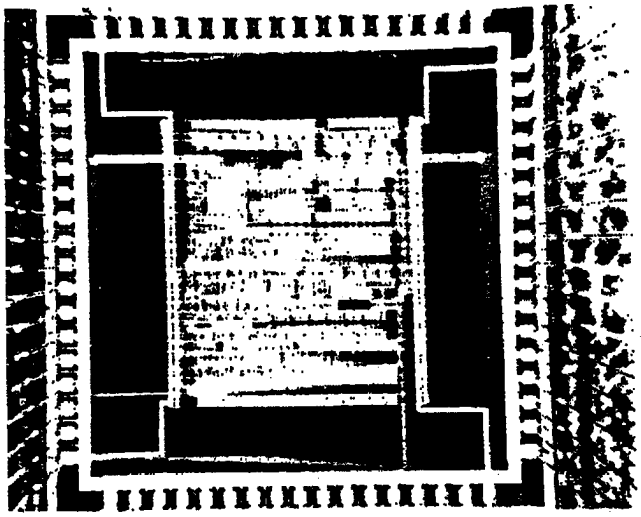

Fig. 14. Photomicrograph of the framer chip.

Serial clock and data inputs enter at the left side, while serial and parallel data outputs and READ strobes exist at the top via buffers for off-chip drive. Access to internal test points is provided at the bottom.

REFERENCES

[1] G. Hayward, L. Linnell, D. Mahoney, and L. Smoot, "A broadband ISDN local access system using emerging technology components," in *IEEE Proc. 1987 Int. Switching Symp.*, pp. A8.1.1–A8.1.5.

[2] L. R. Linnell, "A wide-band local access system using emerging-technology components," *IEEE J. Selected Areas Commun.* vol. SAC-4, no. 4, pp. 612–618, July 1986.

[3] C. Day, J. Giacopelli, and J. Hickey, "Applications of self-routing switches to LATA fiber optic networks," in *IEEE Proc. 1987 Int. Switching Symp.*, pp. A7.3.1–A7.3.5.

[4] R. Boehm, Y. Ching, C. Griffith, and F. Saal, "Standardized fiber optic transmission systems—A synchronous optical network view," *IEEE J. Selected Areas Commun.*, vol. SAC-4, pp. 1424–1431, Dec. 1986.

[5] N. H. Weste, "Virtual grid symbolic layout," in *Proc. 18th Design Automation Conf.*, July 1981, pp. 225–233.

[6] D. G. Boyer and R. R. Cordell, "Structured custom VLSI design methodology," in *Proc. IEEE Int. Conf. Computer Design* (Port Chester, NY), Oct. 1985, pp. 324–325.

[7] H. J. Chao, "Design architectures of a DTDM packet assembler and packet multiplexer," in *IEEE Proc. Montech'87 Conf. Communications*, Nov. 1987, pp. 83–87.

[8] F. E. Owen, *PCM and Digital Transmission Systems*. New York: McGraw-Hill, 1982, pp. 127.

H. Jonathan Chao (S'82–M'85) was born in Taipei, Taiwan in 1955. He received the B.S.E.E. and M.S.E.E. degrees in 1977 and 1980, respectively, from National Chiao Tung University, Hsinchu, Taiwan. In 1985 he received the Ph.D. degree in electrical engineering from Ohio State University, Columbus.

From 1977 to 1981 he was a Senior Design Engineer at Taiwan Telecommunication Laboratories, where he worked on the development of a digital switching system and designed a fault-tolerant controller for the system. From 1982 to 1985 he was a Graduate Research Assistant at Ohio State University, where he designed a VLSI chip with a parallel/pipelined computing structure for robotics applications. Since 1985 he has been at Bell Communications Research, Morristown, NJ, as a Member of Technical Staff, where he has designed two high-speed CMOS VLSI chips for a prototype broad-band local access network. Also since 1985 he has been an Adjunct Assistant Professor in the Electrical Engineering Department of Polytechnic University, Brooklyn, NY, teaching VLSI System Design I and II. Currently he is involved in broad-band ISDN architecture designs and their VLSI implementations.

Thomas J. Robe (M'65) received the B.S.E.E. and M.S.E.E. degrees from the Newark College of Engineering (now the New Jersey Institute of Technology) in 1957 and 1964, respectively.

He served as a Ground Electronics Officer in the U.S. Air Force from 1957 to 1960 at Tyndall Air Force Base, FL. In 1961 he joined the RCA Solid State Division, Somerville, NJ, where he was employed initially as a Linear IC Applications Engineer and subsequently as a designer of linear and digital radiation tolerant IC's for the Trident and MX missile guidance systems. In 1981 he joined Bell Laboratories, Murray Hill, NJ, where he was involved in the design of high-voltage IC's for central-office switching systems. He transferred to Bell Communications Research Inc., at Morristown, NJ, at its inception on January 1, 1984 where his work has concentrated on the design of high-speed electronics hardware components of a broad-band ISDN network prototype. He holds seven U.S. patents.

Mr. Robe was a corecipient of the RCA 1979 David Sarnoff Outstanding Technical Achievement team award for "Development of a Family of High Speed Radiation Tolerant Integrated Circuits."

Lanny S. Smoot (M'85) was born in Brooklyn, NY, in 1955. He received the B.S. and M.S. degrees in electrical engineering from Columbia University, New York, NY, in 1977 and 1978 respectively.

From 1978 to 1983 he was a Member of Technical Staff in the Loop Systems Laboratory at Bell Laboratories where he was involved with the development of various loop transmission systems including the optical receiver designs for some of the first fiber-optic carrier systems. He was promoted to his current position in 1983 and transferred to Bell Communications Research, Morristown, NJ, where he has been responsible for research on hardware aspects of the Broadband Integrated Service Digital Network. He is currently District Manager of the Broadband Network Technologies Research Group at Bell Communications Research. He holds several patents in the fields of subscriber carrier systems, optical receiver design, and optical local area networks.